

- 
- Evolución histórica de la computadora
  - Máquina de Von Neumann
  - Buses\*
  - Memoria
    - Cache\*
  - Entrada / Salida\*
  - Conversión A-D y D-A\*

\*Se agregaron algunas cosas de otros resúmenes de cubawiki

## La organización y la arquitectura

El estudio de la organización del computador se centra en cómo funciona la computadora **físicamente** (*señales de control, tipos de memoria y demás*) mientras tanto El estudio de la arquitectura del computador se encarga de la **estructura y el comportamiento**, la lógica y aspectos más abstractos de la computadora por ejemplo: *Tipos de datos, tipos de direccionamiento, tipos de registros, etc.* La organización y la arquitectura no son temas independientes, se complementan entre sí. La arquitectura de una computadora se da en una combinación entre el hardware y la **ISA** (instruction set architecture) que es lo que se utiliza para poder comunicarse con el hardware, con ella podemos correr todo el software sobre una máquina. Una computadora se compone principalmente por:

- El procesador que interpreta y ejecuta programas
- La memoria donde se guardan los datos y los programas
- Un mecanismo para transferir datos del exterior

## Historia

La historia del desarrollo de las computadoras se divide en generaciones en base a la tecnología usada en cada una.

### Generación cero: calculadoras mecánicas (1642-1945)

Por los años 1500, las personas utilizaban abacos para calcular y guardar el resultado en **numeros romanos**, luego cuando llegó el sistema decimal se crearon distintos dispositivos para poder hacer cálculos más exactos y rápidos. A **Wilhelm Schickard** (1592-1635) se le acredita la creación de la primera calculadora mecánica la cual podía sustraer y agregar

---

---

numeros. luego **Blaise Pascal** (1623-1662) creo la Pascaline para ayudar a su padre con la contaduria, esta podia sumar con carry y sustracción. Hubo muchas similares a la Pascaline hasta que **Charles Babbage** (1791-1871) contruyo una maquina diferencial la cual fue diseñada para mecanizar la solucion de funciones polinomicas, luego en 1833 creo la "The Analytical Engine" la cual fue diseñada para ser mas versatil que su antecesora tambien iba a tener varios componentes similares a los que tienen las computadoras actuales. **Ada Lovelace** fue quien escribio como deberian calcularse los numeros, por ello se la considera la primera programadora.

## Primera generación: Computadoras de tubos de vacio (1945-1953)

**konrad Zuse** (1910-1995) basandose en la "The Analytical Engine" de Babbage creo la Z1 la cual utilizaba **releys electromecanicos** y era posible programarla, como no habia mucho dinero en alemania en ese momento, se utilizaron films de peliculas descartas para hacer inputs a la maquina y **no tenia tubos de vacio** aunque fue diseñada para tenerlos, hubo otras maquinas creadas pero gracias a la guerra no se avanzo

**John Atanasoff** (1904–1995) es a quien se le atribuye la creacion de The Atanasoff Berry Computer (ABC) la cual fue una maquina de tubos de vacio binaria porque fue construida para calcular sistemas de ecuaciones lineales, unos años mas tarde John **Mauchly** (1907–1980) y **J. Presper Eckert** (1929–1995) crearon la ENIAC (Electronic Numerical Integrator and Computer) la cual fue multiproposito y **completamente digital**, usaba unos 17.468 tubos de vacio y podia guardar unos 1000 bits. Fue financiada por la armada de estados unidos en la segunda guerra mundial ya que facilitaria calcular la trayectoria de misiles.

## Segunda generación: Computadoras de transistores (1954-1965)

Los tubos de vacio se solian quemar con facilidad y era necesario reemplazarlos demasiado seguido. En 1948 los investigadores de **Bell Laboratorios** inventaron los **transistores** los cuales consumian menos energia que los tubos de vacio ademas de ser mas pequeños. Esto revoluciono la construcción de computadoras, **IBM, DEC y Univac** crearon a la "7094" para aplicaciones cientificas y la "1401" para negocios. Despues **CDC** creo la primera super computadora: la CDC 66000 la cual podia usar 10 millones de instrucciones por segundo.

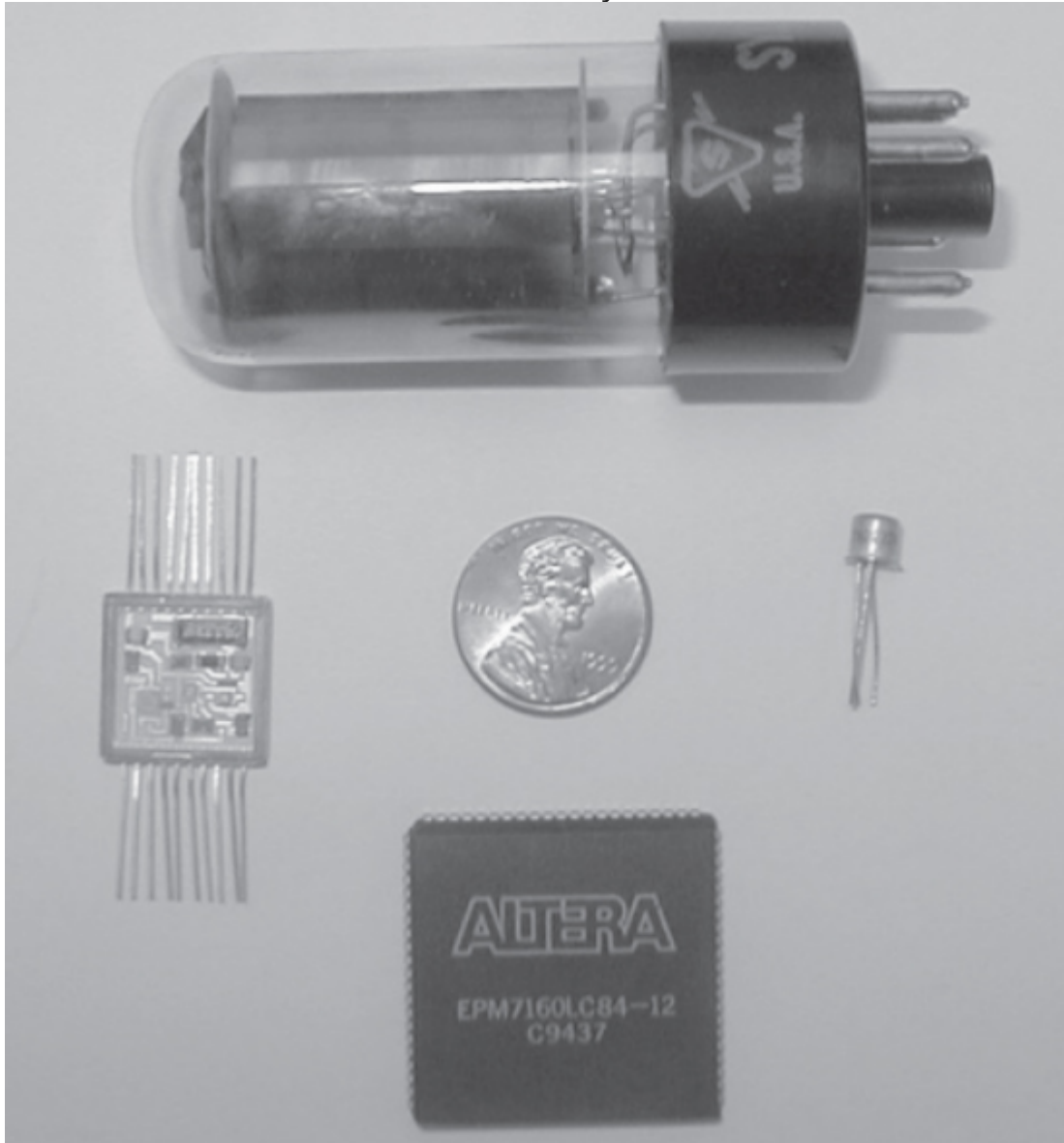
## Tercera generación: Computadoras de circuitos integrados (1965-1980)

**Jack Kilby** invento los circuitos integrados (IC) o tambien llamados microchips los cuales eran mucho mas pequeños y podian contener a los transistores, por lo que las

---

---

computadoras se volvieron mas pequeñas y rapidas, en este momento IBM empezo a comercializar computadoras las cuales **usaban el mismo assembler** lo que hacia el software mas compatible entre ellas tambien fue necesario crear sistemas operativos que permitan que las computadoras **sean usadas por mas de una persona a la vez** para abaratar costos una de ellas fue la PDP-8 y PDP-11.



Comparación entre un tubo de vacio, un transistor, un chip y un circuito integrado

## Cuarta generación: Computadoras VLSI (1980 - ????)

En la tercera generación multiples transistores entraban en un circuito integrado, mientras avanzaban las tecnicas de produccion la cantidad de transistores aumentaba. Ahora hay varios niveles de integracion: SSI (small scale integration): 10 a 100 componentes por chip, MSI (medium scale integration): 100 a 1000 componentes por chip, LSI (large-scale integration) 1000 a 10.000 componentes y por ultimo: VLSI. **La cantidad de transistores por**

---

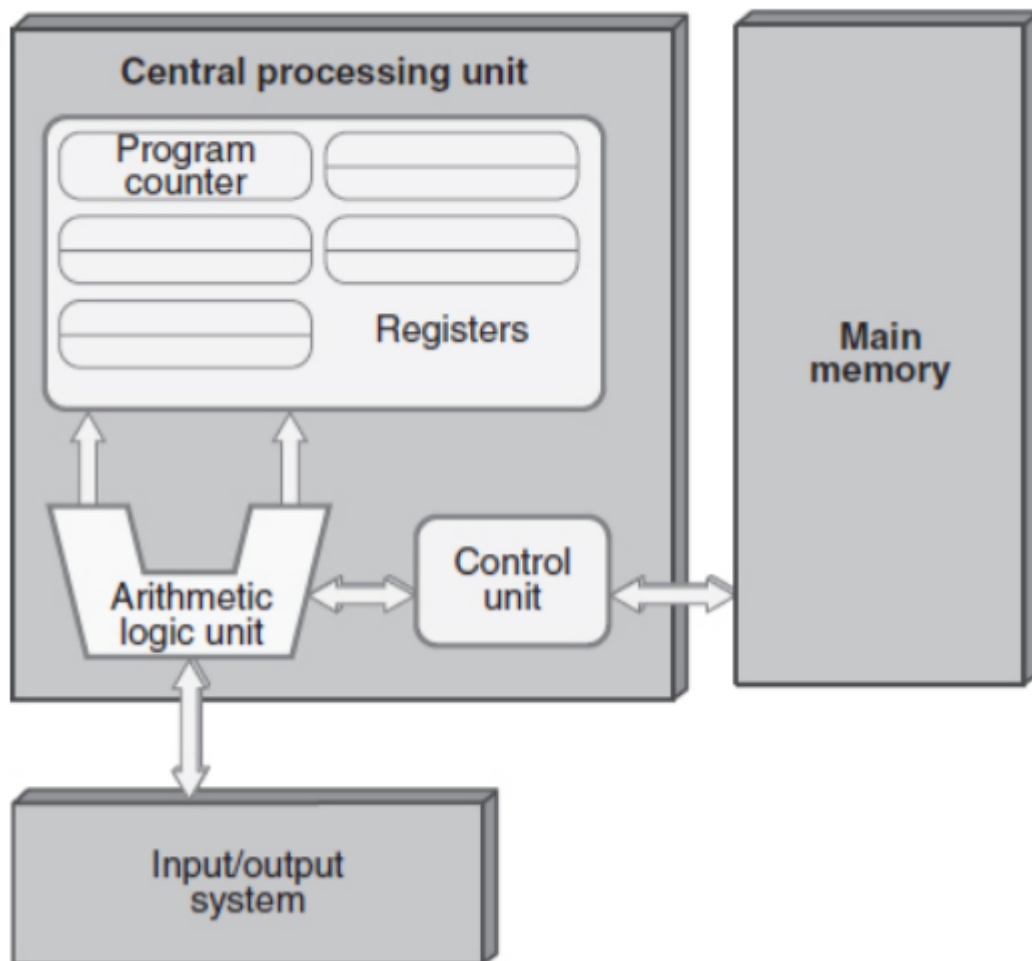
---

**chip va creciendo constantemente.** En 1971 **intel** creo el primer microprocesador: el 4004 el cual corria a 108KHz, luego se creo la RAM (random access memory), todo esto causo que las computadoras puedan ser accesibles para el publico general, en 1975 la Altair 8800 fue la primera microcomputadora, creada por **MIT**. luego siguieron apple I entre otras, despues en 1981 **IBM** introdujo **la PC (Personal Computer)** lo que se volvio un standard en la industria con el tiempo.

## Modelo de Von Neumann

En las computadoras electronicas, **programar era sinonimo de conectar cables** lo que lo volvia algo mas similar a hacer ingenieria electronica que a pensar algoritmos, antes de la ENIAC estuviera completa, **John W. Mauchly y J. Presper Eckert** se les ocurrio una forma de almacenar las intrucciones con lo cual no tendrian que reconstruir el sistema por cada problema a resolver, la idea no pudo ser publicada enseguida ya que la ENIAC era un proyecto secreto y ellos estaban involucrados. **John von Neumann** quien fue un matematico húngaro, profesor de matematicas en la universidad de Princeton y the Institute for Advanced Study (IAS) el cual trabajo en la periferia del ENIAC luego de leer la idea decidio publicarla y publicitarla construyendo la computadora IAS. Actualmente todas las computadoras con programas almacenados utilizan la arquitectura Von Neumann, todas tienen las siguientes características:

- Principalmente consta de una central processing unit (CPU) con una control unit, una arithmetic logic unit (ALU), registers (pequeños almacenamientos), un program counter, una memoria principal donde se guardan los programas que controlan las operaciones de la computadora y un sistema de entrada y salida.
  - Capacidad de procesar instrucciones de una forma **secuencial**.
  - Los datos y las instrucciones de un programa se guardan en la misma memoria y la ejecución del programa depende del estado de la memoria.
  - Tiene **una sola ruta física o lógica** de conexión entre la memoria principal y el CPU forzando la alteración entre ciclos de instrucciones y ejecuciones.
-



El I/O pasan por el acumulador que esta dentro de la ALU, esta arquitectura corre programas siguiendo el **ciclo fetch-decode-execute**. La forma en la que trabaja es la siguiente:

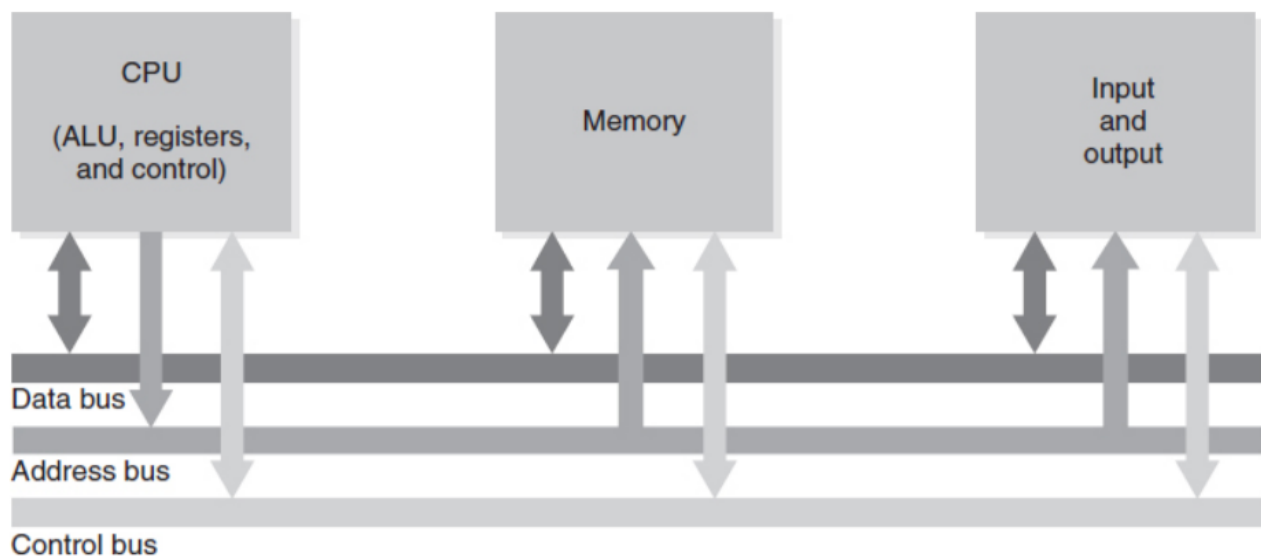
1. **Control unit** fetchea la proxima intruccion del programa desde la memoria, utiliza el program counter determina donde esta la instruccion.
2. La instruccion fecheada se **decodifica** de tal forma que puede ser entendida por la ALU.
3. Los datos necesarios para operar se buscan en la **memoria** y se fetchean en los **registros**.
4. La **ALU ejecuta** la instruccion y guarda el resultado en los registros o memoria.

Luego esta idea se extiende a almacenamientos con **acceso lento** (por ejemplo: discos duros) la cual es copiada a almacenamiento de **acceso rapido** (por ejemplo la RAM) para una proxima ejecucion, la forma en que esto funciona se simplifico en algo llamado modelo de sistema de bus que es la forma en la que se mueve la informacion entre los distintos tipos

---

---

de almacenamiento



## Componentes del Modelo Von Neumann

Los siguientes se encuentran dentro del CPU:

- ALU (Arithmetic Logic Unit) es un circuito digital electrónico que realiza las operaciones aritméticas y lógicas bit a bit en números binarios enteros.
- Control Unit es un componente que dirige las operaciones del procesador. Le dice a la memoria, ALU y los dispositivos de entrada y salida como responder a las instrucciones de un programa. El cual contiene:
  - IR almacena la instrucción que se está ejecutando actualmente o está siendo decodificada
- Registros son unidades de almacenamiento pequeñas que son típicamente dirigidas por mecanismos distintos de la memoria principal y a los que se puede acceder más rápido. Dentro de los SPR (special purpose registers) se encuentra:
  - PC indica en qué parte del programa está la secuencia de un programa. Aumenta luego de hacer fetch de una instrucción.
- Memoria que se utiliza para almacenar tanto datos como instrucciones
- Mecanismos de entrada y salida de esta forma podemos ingresar o extraer datos de la máquina.

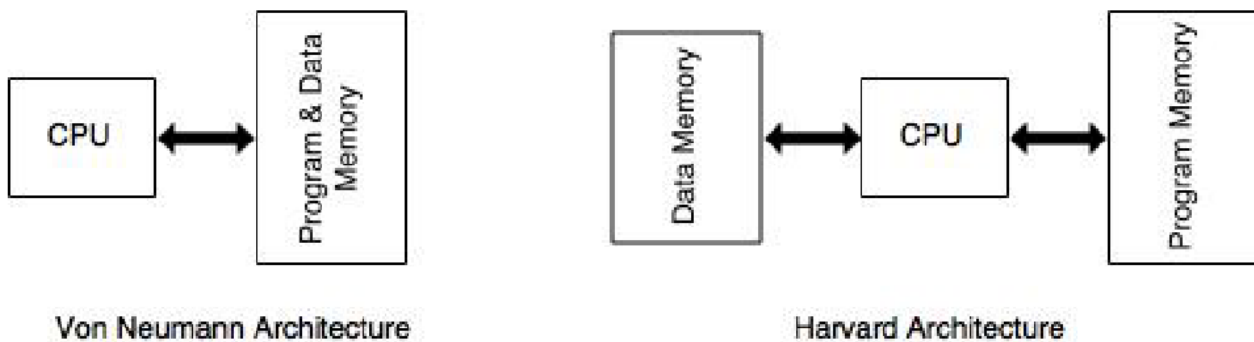
## El modelo de harvard

---

---

A diferencia del Von Neumann tiene un conjunto dedicado de direcciones y buses de datos para leer datos desde memoria y escribir datos en la misma y otro conjunto de direcciones y buses de datos para buscar instrucciones.

---



## Buses

La CPU necesita comunicarse con otros componentes por lo cual existe el bus que cumple esa función, el cual es un **conjunto de cables** que conectan todo el sistema, los cables siguen el movimiento paralelo de los bits. El hecho de conectar todo el sistema hace fácil conectar más dispositivos si es necesario, pero como el bus **solo puede ser utilizado por un dispositivo a la vez** se genera un cuello de botella en la comunicación, también la velocidad del bus es afectada por la distancia entre componentes ya que implica cables más largos, por esto los dispositivos conectados al bus se dividen en categorías: Master (el dispositivo que inicia la acción) y Slave (es aquel dispositivo que responde los requisitos del Master).

## Transferencia de datos en un bus dedicado:

en un bus dedicado cada línea tiene un propósito

- **Escritura (master a slave):** en un ciclo de clock master envía la dirección + datos por buses distintos.
- **Lectura (slave a master):** en un ciclo de clock master envía la dirección por bus de direcciones+slave coloca el dato en el bus de datos.

## Transferencia de datos en un bus multiplexado:

en un bus multiplexado, las líneas tienen propósitos diferentes en distintos instantes de tiempo (Ej.: bus de datos / dirección según una línea de control).

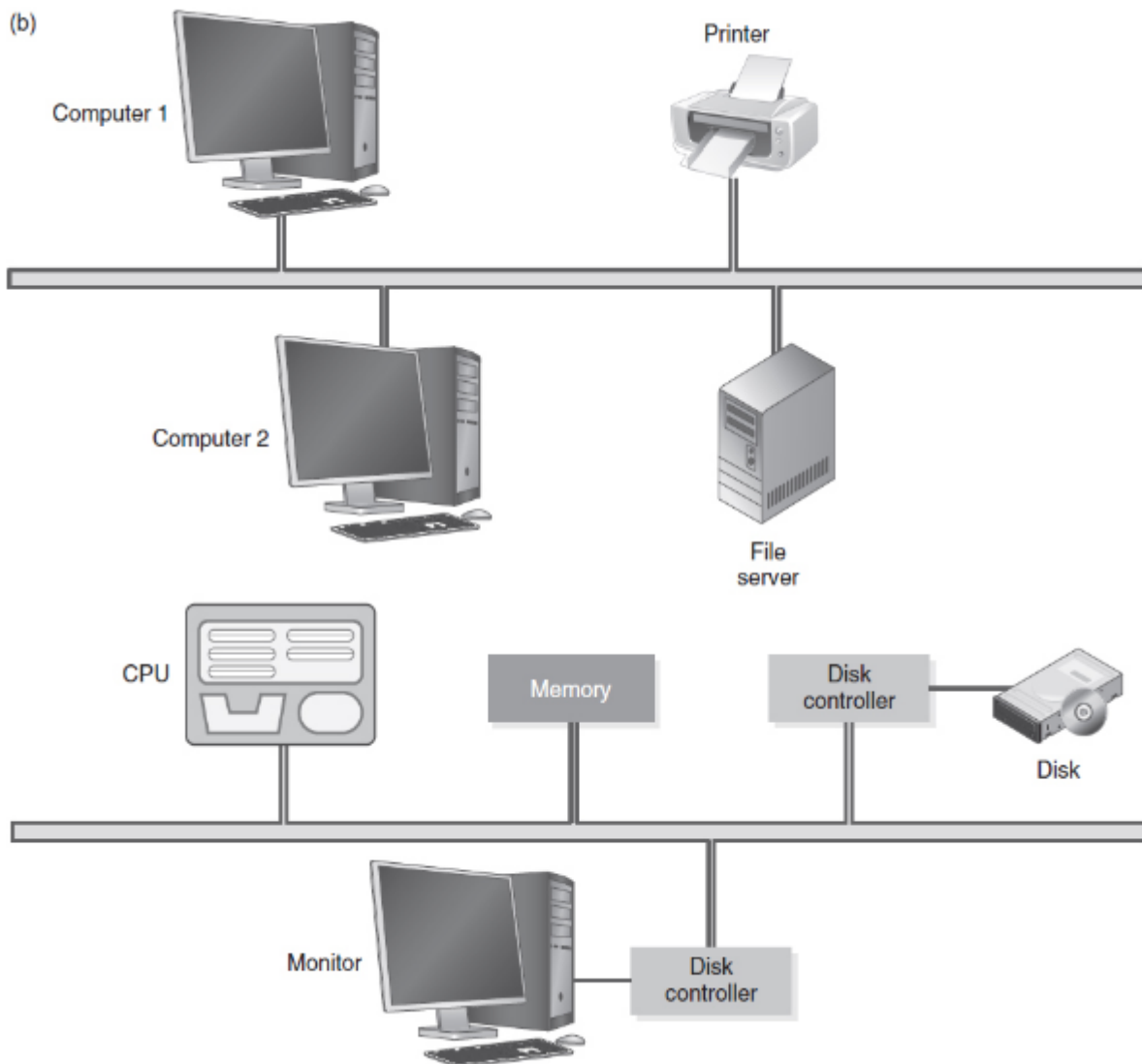
- **Escritura:** transmisión de dirección + transmisión de dato
-

- 
- **Lectura:** transmisión de dirección + espera que slave coloque dato (transferencia de bloques de datos: dirección + varios ciclos de datos)

El bus puede ser "punto por punto", conectando específicamente dos componentes



o puede ser un camino de datos común el cual conecta varios dispositivos, los cuales requieren compartirlo.

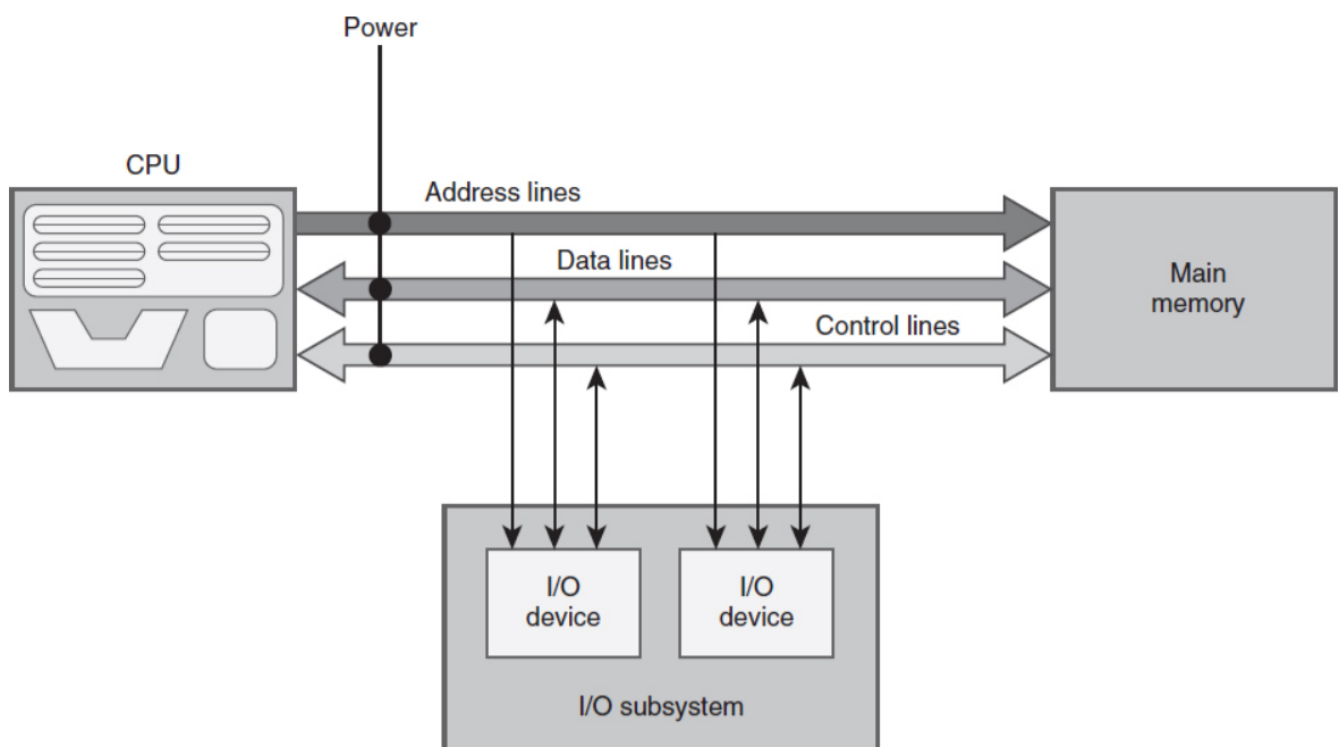




---

Cuando se comparte el bus, es necesario tener un **protocolo de bus** el cual define las reglas de uso del bus.

En los casos del grafico anterior necesitaremos líneas de datos las cuales se llaman "Data bus" que contienen la información actual la cual va a ser movida a otro lugar, luego las Líneas de control las que indican cual es el dispositivo que tiene permiso para utilizar el bus y para que lo va a utilizar (*por ejemplo: leer o escribir memoria*), también se encarga de **pasar las señales necesarias**, después están las líneas de dirección las cuales indican la localización de los datos a ser leídos u escritos (*por ejemplo en memoria*), por último las líneas de poder se encargan de proveer la electricidad necesaria.



Normalmente las transacciones del bus son para enviar direcciones (para leer o escribir), Transferir datos a memoria desde registros (memoria a escribir), también los buses son utilizados por I/O para escribir y leer. Toda transacción del bus ocurre dentro de **un ciclo el cual es marcado por 2 ciclos de clock**.

Los buses se dividen en diferentes tipos ya que existen distintos tipos de información a transportar y diferentes dispositivos que utilizan los buses, esas categorías son:

- Processor-memory buses: son aquellos con alta velocidad y se encuentran más cerca del procesador y está emparejado con la memoria, generalmente tienen un diseño específico.
  - I/O buses: generalmente son más largos y pueden ser usados por varios dispositivos y también son compatibles con varias arquitecturas.
-

- 
- Backplane bus: construido en el chasis de la maquina y conectado al procesador, la memoria, dispositivos I/O.

Las **computadoras personales** tienen un bus interno (llamado system bus) que conecta el procesador, la memoria y otros dispositivos internos y luego tienen un bus externo con el cual se conectan los dispositivos externos. Tambien los buses se separan entre asincronicos (los cuales no estan sujetos al clock, los eventos que suceden provocan otros eventos) y sincronicos (los cuales tienen ciclos siguiendo los ciclos del clock).

## Memoria

Las computadoras que siguen el modelo de Von Neumann, se centran en la memoria, por lo que es un componente bastante importante el cual tiene varios tipos porque **intenta de alguna forma seguir el avance de los procesadores** para no generar cuellos de botella.

## Tipos basicos de memoria

- Volatiles: son aquellas que requieren energia para mantener la informacion almacenada; conserva sus contenidos mientras esta encendida y cuando se apaga los datos se pierden inmediatamente o muy rapidamente
- No Volatiles: Aquellas que pueden recuperar la informacion almacenada, incluso despúes de haber sido desconectadas.

## Principales tipos de memoria volatil

- Memoria cache: Es una memoria pequeña y rapida (pero mas cara) generalmente sirve para tener a **mas accesibles los datos** que se estan utilizando mas frecuentemente.
- RAM (random-access memory): La cual en realidad es una memoria que se puede **leer y escribir**, es usada para guardar los datos y los programas que esta por ejecutar la computadora, esta memoria es **volatil** lo que quiere decir que cuando no tenga energia todo lo almacenado se borrara.

Por lo general se utilizan dos tipos de chips para construir la RAM y la cache:

- DRAM (dynamic random-access memory): es construida con pequeños capacitores que van perdiendo electricidad, por lo que requiere una recarga cada muy poco tiempo a pesar de eso usa **menos energia y es condensada (entran mas bits por chip)**. generalmente se usa para la RAM
  - SRAM (static random-access memory): es construido con circuitos similares a los D flip-flops los cuales mantienen la informacion mientras haya energia y es mas **rapida** pero mas cara (generalmente se usa para la cache).
-

---

Ademas algunas computadoras tienen alguna pequeña ROM donde tienen guardada alguna **informacion critica** para el funcionamiento del sistema, esta memoria no es volatil por lo que la informacion guardada en ella no se borrara aunque no haya energia. esta memoria se utiliza en sistemas donde no es necesario cambiar la programacion (*por ejemplo: en juguetes, calculadoras y impresoras*)

Existen varios tipos de ROM:

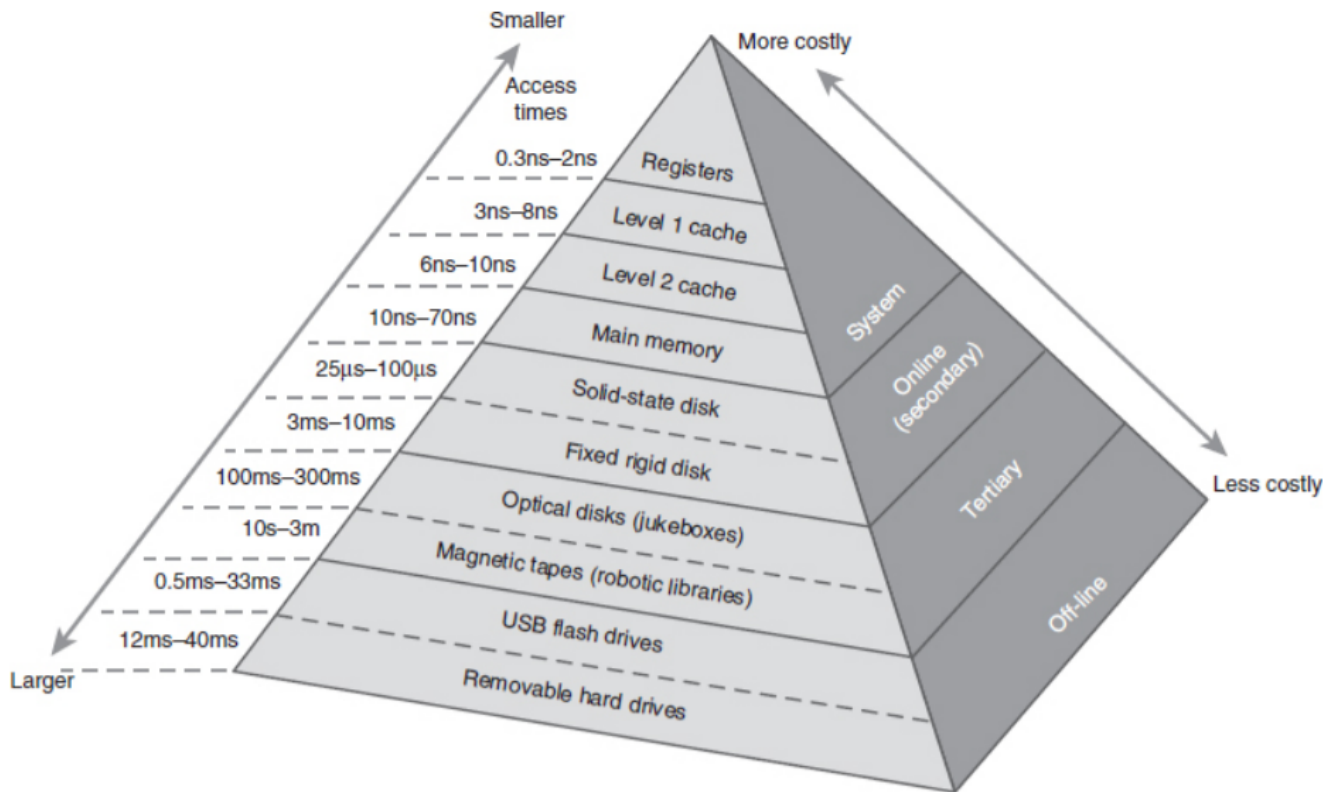
- PROM (programmable read-only memory): Es una memoria que puede ser programada teniendo las **herramientas especiales**, pero una vez programada **no se puede modificar**, ya que se queman los fusibles dentro de la memoria.
- EPROM (erasable PROM): Es una memoria programable y **reprogramable** (para ello se necesita **borrar todo el chip** con luz ultra violeta).
- EEPROM (electrically erasable PROM): Es una memoria que **no requiere herramientas especiales** para borrar la memoria, solo se necesita aplicar electricidad y no es necesario borrar todo el chip
- Flash memory: Es una memoria igual a la EEPROM pero con la posibilidad de **borrar o escribir en bloques**, lo que la hace mas rapida que la EEPROM se utiliza en diferentes dispositivos por ejemplo: *celulares y camaras*.

## Jerarquia de la memoria

Como no todas las memorias son igual de rapidas, baratas y eficientes las computadoras utilizan combinaciones de ellas intentando obtener el **mejor rendimiento al menor costo** posible para eso existe la jerarquia, la cual es una regla: cuando mas rapida sea la memoria mas caro sera cada bit guardado en ella.

Las memorias que constituyen la jeraquia normalmente son:

- Registros: se encuentran dentro del procesador
  - Cache: es una memoria rapida donde se guardan temporelmente los datos usados con frecuencia, esta conectada a una memoria principal.
  - Memoria principal: normalmente es una memoria con velocidad media y generalmente se complementa con la memoria secundaria.
  - Memoria secundaria: generalente se compone de un disco duro (*por ejemplo un HDD*) el cual contiene datos que no es accesible de forma directa para el CPU por lo cual estos datos necesitan pasar por la memoria principal para ser accedidos
  - Otras: son aquellos que requieren una intervencion fuera del sistema para ser conectados (*por ejemplo: un pendrive*), los datos para ser accesibles necesitaran ser transferidos a la memoria secundaria.
-



Esta jerarquia tambien habla sobre la distancia que hay entre la memoria y el procesador, la cual **se mide con la cantidad de ciclos de la maquina** necesarios para poder acceder a la memoria. Cuando mas rapida es una memoria esta mas cerca del procesador pero es mas pequeña.

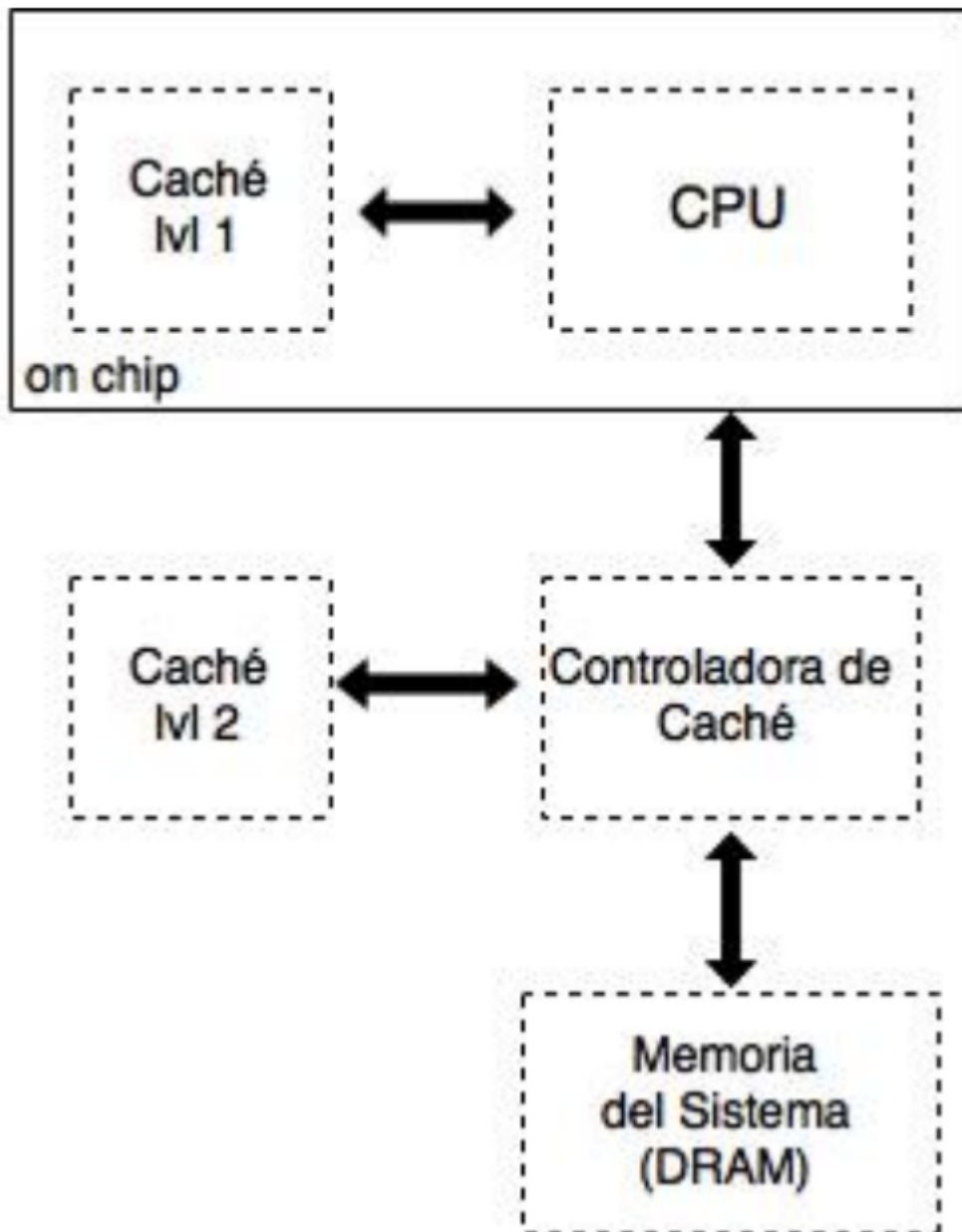
## Cache

El procesador de una computadora esta constantemente leyendo datos de la memoria pero tiene que esperar ya que la memoria es mas lenta. La memoria cache es una memoria muy rapida pero es pequeña y el procesador la necesita con mucha frecuencia. *Por ejemplo: una caja de herramientas pequeña es como una cache para un carpintero ya que asi puede tener a mano las herramientas que mas utilizara en cierto momento.*

La computadora realmente no sabe cuales seran los datos proximos a utilizar por lo que la controladora de la cache usa el principio de localizacion y por cada vez que es necesario acceder a la memoria principal se transfiere un bloque a la cache. La localizacion del bloque dependera de la politica de mapeo de la cache y el tamaño de la cache que puede variar enormemente, generalente en una PC, el nivel 2 (L2) es mas grande y reside entre el procesador y la memoria principal mientras que el nivel 1 (L1) se encuentra dentro del

---

procesador (tambien a veces se incluye L3).



## Algunos terminos

- Hit: se refiere a cuando el procesador busca cierto dato en una dirección de memoria y efectivamente ese dato esta alli.
  - Miss: es cuando el procesador lee la memoria y no encuentra el dato buscado.
  - Hit rate: se trata del porcentaje de hits de una memoria cache.
  - Miss rate: es el porcentaje de miss de la memoria cache.
  - Hit time: se refiere al tiempo necesario para conseguir un hit.
-

- 
- Miss penalty: es el tiempo que se requiere para procesar un miss, generalmente se requiere mas tiempo que para un hit. porque cuando ocurre un miss el procesador debe recurrir a la memoria principal y la demora en el acceso hace que el pipeline se atasque (stall). Una vez recuperado el dato de memoria, se requieren varios ciclos de clock para recuperar el ritmo de operación del pipeline.

El pipeline es lo que permite superponer en el tiempo la ejecución de varias instrucciones a la vez. (No requiere hardware adicional, sino lograr que todas las partes del procesador trabajen a la vez) este funciona con el concepto de una linea de montaje: cada operacion se descompone en partes y se ejecutan al mismo tiempo partes diferentes (stages) de diferentes operaciones.

## Principio de localizacion

Como sabemos que la memoria normalmente se lee secuencialmente (si es que no hay un salto) normalmente conviene transferir un bloque entero, ya que probablemente se usen en un futuro cercano

## Formas basicas de localidad

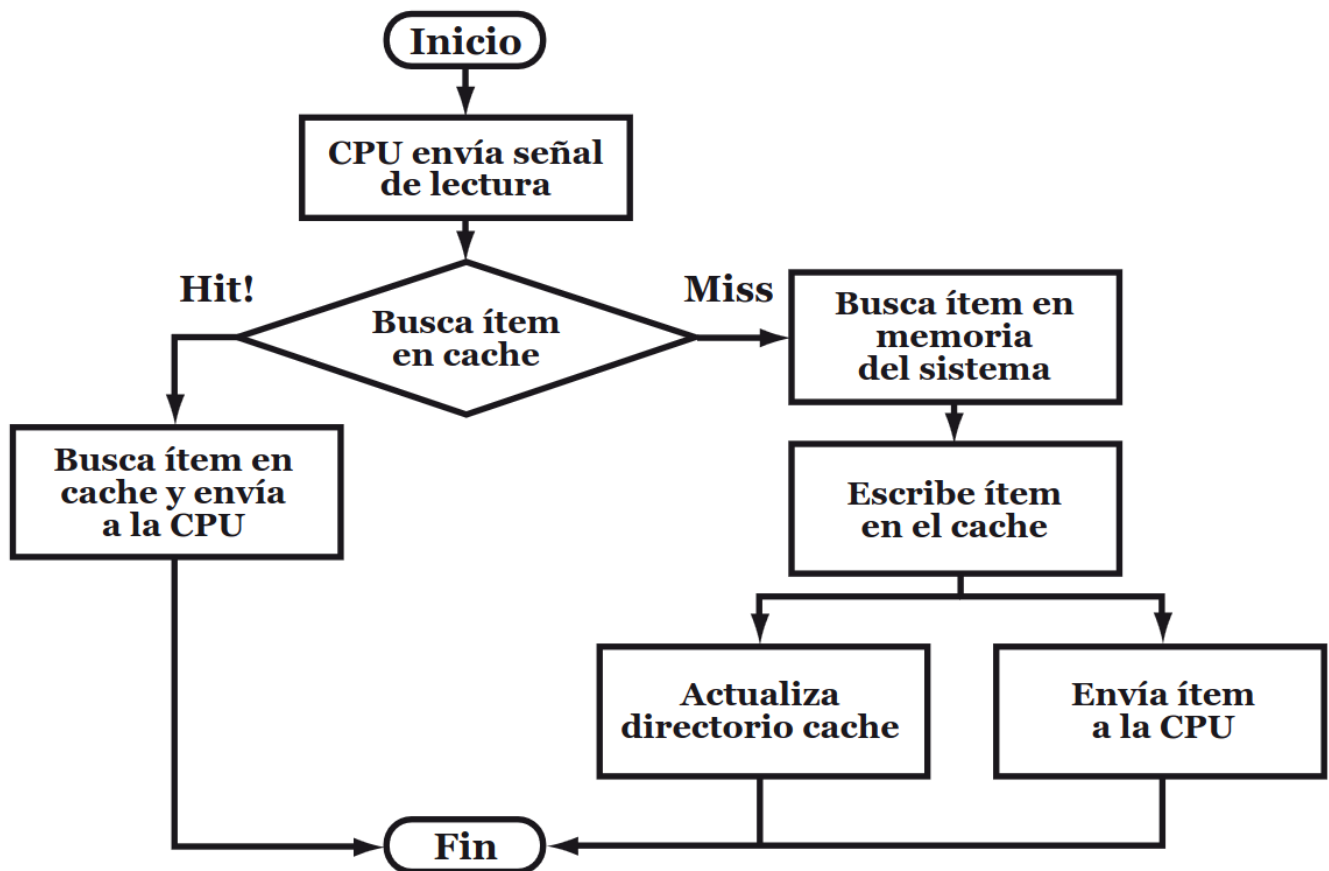
- Localidad temporal : los elementos accedidos recientemente probablemente sean accedidos nuevamente pronto.
- Localidad espacial: Si un ítem es referenciado, es altamente probable que se referencie a los ítems vecinos a éste (*por ejemplo en arrays y loops*).
- Localidad secuencial: las intrucciones tienden a ser accedidas de forma secuencial.

La cache a diferencia de la RAM normalmente esta construida con SRAM por lo que es accesible en 10ns mientras que DRAM (la que se suele utilizar para la RAM) tarda 50ns, tambien sucede que la cache no es accedida en base a una direccion sino que se accede en base al contenido, para simplificar el proceso de saber si cierto contenido esta en la cache

---

---

se utilizan algunos algoritmos de mapeo de cache.



**Linea** Elemento minimo de palabra de datos dentro de la cache. Corresponde a un multiplo del tamaño de la palabra de datos de memoria.

## Esquemas de mapeo de cache

Para que la cache funcione debe guardar la informacion que va a usar el CPU proximamente. Pero sabemos que la CPU genera una direccion de la memoria principal, asi que para saber en que parte de la cache se encuentran los datos copiados de esa direccion utiliza los esquemas para pasar de una direccion a una localizacion. **Tag**: es un directorio de cache, el cual guarda tiras de bits que caracterizan las líneas que tenemos almacenadas en la cache.

## Esquemas de mapeo

- Mapeo directo: consiste en dividir la memoria principal en páginas del tamaño de la cache. Luego, cada línea de cada la página tiene un lugar específico en la cache, y por lo tanto, solo tenemos que almacenar como tag, en el directorio de la cache, la página a la que pertenece el dato.
-



← Bits in Main Memory Address →

Para recuperar el dato, únicamente se necesita buscar la línea en la que debería estar guardado, y revisar si tiene el tag correspondiente, y en caso de que coincidan, indexarlo y obtener el dato. este metodo es barato ya que no requiere hardware adicional pero si llega un nuevo dato, y la posición correspondiente en la cache ya estaba ocupada, se elimina la línea que estaba allí anteriormente, y por lo tanto no podemos almacenar una misma línea de distintas páginas. Por ejemplo, no podemos tener la línea 5 de la página 3, y al mismo tiempo la línea 5 de la página 10, a pesar de tener espacio libre en la cache.

- Mapeo totalmente asociativo: permite que cada línea de memoria principal se coloque en cualquier lugar de la cache. Luego, para poder identificar mediante el tag cada dato, necesitamos saber a qué página pertenece, y también a qué línea dentro de la página.

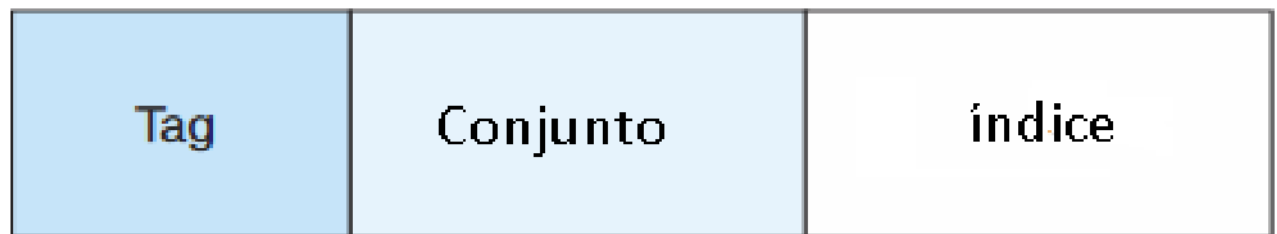


← Bits en una dirección de memoria →

Al momento de hacer una lectura, hay que comparar con todos los tags del directorio, ya que la línea puede estar en cualquier parte de la cache, y por lo tanto es más lento (o requiere de un HW específico para poder realizar todas las búsquedas al mismo tiempo, resultando en un esquema más costoso).

- Mapeo asociativo por conjuntos: usamos la dirección para mapear el bloque a una determinada ubicación de la cache, cada dirección se mapea a un conjunto de N vías de cache.
-





← Bits de una dirección de memoria →

Cuando hacemos una búsqueda de un dato en la cache, solo tenemos que revisar las N vías correspondientes a al dato. tiene un mayor costo (por el HW específico para realizar búsquedas en simultáneo) y el tamaño de la vía es N veces más chica que la línea en Mapeo Directo (para un mismo tamaño de cache).

## Políticas de reemplazo de contenido

En el esquema de mapeo directo no es necesario pero si en los asociativos porque es necesario tener un algoritmo de reemplazo para determinar qué línea de la cache hay que reemplazar por la nueva. Cuando trabajamos con un esquema completamente asociativo, se podría reemplazar cualquiera de las K líneas de la cache, mientras que con esquema de N vías, tenemos solo N opciones.

- Least Recently Used (LRU) suponiendo que un dato que fue accedido recientemente es más probable que se vuelva a necesitar que uno que no. por lo que se hace un seguimiento de la última vez que se accedió a cada línea de la cache, guardando el timestamp en el directorio, y remover aquella línea (o vía) que se haya utilizado menos recientemente. (esto relentiza la cache y ocupa espacio significativo).
  - Least Frequently Used (LFU) consiste en que el sistema lleve un registro del número de veces que se hace referencia a una línea en la memoria. Cuando la cache está llena, y requiere más espacio, el sistema removerá la línea (o vía) con la frecuencia de referencia más baja. se implementa asignando un contador a cada línea (o vía) que se carga en la cache. Cada vez que se hace una referencia a esa línea, el contador aumenta en uno. Cuando la cache está llena, y tiene un nueva línea que debe ser guardada, el sistema buscará la línea (o vía) con el contador más bajo, y lo reemplazará por la línea entrante, reiniciando el contador.
  - First In - First Out (FIFO) con este algoritmo la línea (o vía) que ha estado en la cache por más tiempo sería la próxima en ser reemplazada.
  - Random El problema con LRU y FIFO es que hay situaciones en las que pueden causar trashing, es decir qu tiran constantemente una línea, luego la traen de vuelta, repetidamente. Algunas personas argumentan que el reemplazo random, aunque a veces arroja datos que se necesitarán pronto, nunca genera trashing.
-

---

Desafortunadamente es difícil tener un reemplazo verdaderamente random, y además puede disminuir el rendimiento promedio

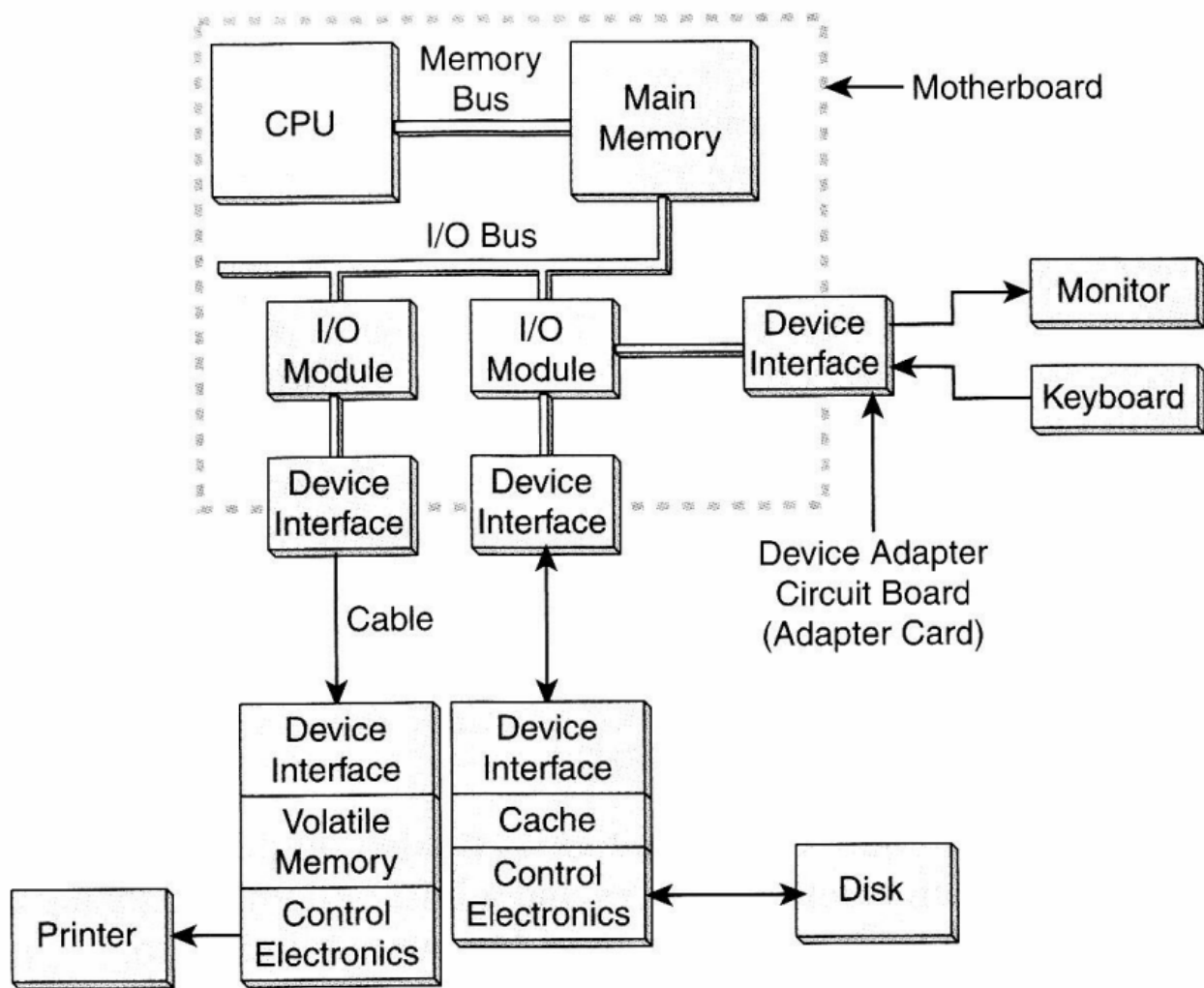
## Políticas de escritura

Una variable que esta en cache tambien esta alojada en alguna direccion de la memoria principal (ambos valores deben ser iguales). Cuando el procesador los modifica hay varios modos de actuar:

- Write through: El procesador escribe en la memoria principal y el controlador de cache actualiza el cache con el nuevo dato.
- Write through bufferd : El procesador actualiza la cache y el controlador cache, luego actualiza la copia en memoria principal mientras el procesador continua ejecutando instrucciones y usando datos de la memoria cache.
- Copy back: Se marcan las lineas de la memoria cache cuando el procesador escribe en ellas. Luego en el momento de eliminar esa linea del cache el controlador cache debera actualizar la copia de la memoria principal. Si el procesador realiza un miss mientras el controlador de cache esta accediendo a la memoria principal para actualizar el valor, debera esperar hasta que el controlador de cache termine la actualizacion para recibir desde este la habilitacion de las lineas de control.

## Entrada y salida

Es la comunicacion entre la computadora y un humano u otro sistema de procesamiento de informacion. Los Inputs son las señales o datos recibidos por el sistema y outputs son las señales o datos enviados por este. Por ejemplo, lectura de datos desde un disco externo es una operacion de I/O.\*



De esto se encarga el subsistema de E/S el cual esta conformado por estos elementos:

- Espacios de memoria principal dedicados a funciones de E/S
- Buses que permiten mover datos dentro y fuera del sistema
- Módulos de control de computadora y de dispositivos periféricos
- Interfaces a componentes externos (como teclados y discos)

Los módulos de E/S se encargan de mover datos entre la memoria principal y una interfaz de dispositivo en particular. se debe asegurar de que los dispositivos estén listos para el siguiente bloque de datos, o que la computadora esté lista para recibir el próximo bloque de datos provenientes del dispositivo periférico.

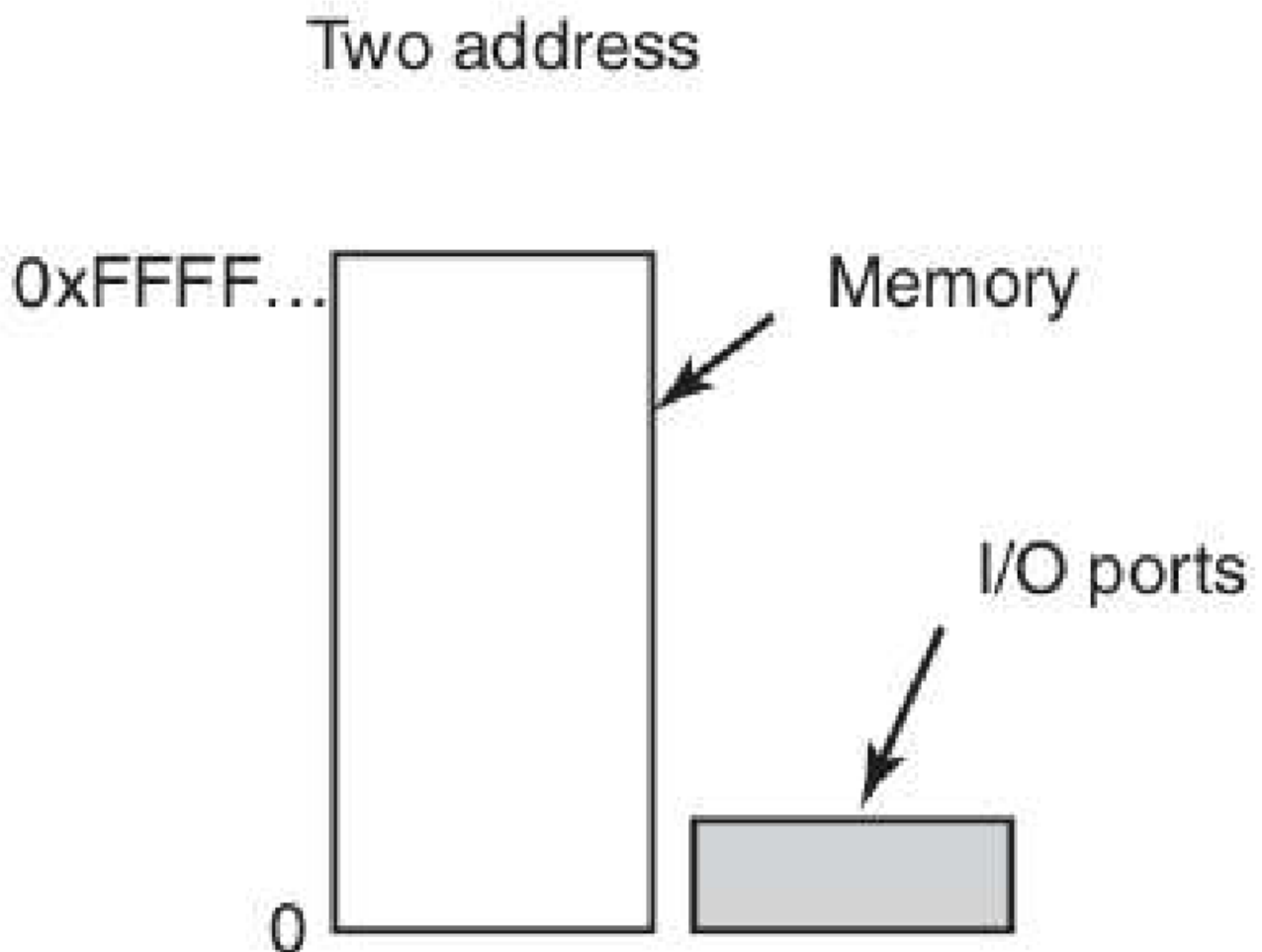
## Métodos de acceso a E/S

---

Los dispositivos tienen un par de registros que utilizan para comunicarse con la CPU. Escribiendo en esos registros, el SO puede enviar comandos al dispositivo para que realice alguna acción (transferir datos, apagarse, prenderse, etc). Leyendo de esos registros, el SO puede saber cuál es el estado del dispositivo, si es que está preparado para aceptar un nuevo comando, etc.

## formas de comunicación

- Puertos dedicados: cada registro de control tiene asignado un puerto de E/S, un entero de 8 o 16 bits. El conjunto de todos los puertos de E/S forman el I/O port space, que es un espacio de E/S protegido para evitar que los programas de nivel usuario puedan acceder al mismo. En este esquema de acceso a dispositivos de E/S, los espacios de memoria y el espacio de E/S son distintos:



- Mapear a Memoria: se mapean en la memoria principal todos los registros cada registro de control es asignado a una única dirección de memoria, que no puede ser utilizada por el resto del sistema. En general, las direcciones asignadas suelen estar en alguno de los extremos, es decir o bien en las direcciones bajas o bien en las altas.
-

---

**Registros de E/S**



**Memoria**



## Métodos de control de E/S

Normalmente los dispositivos tienen un registro de estado, el cual cambia cuando se escribe sobre el registro de datos a not ready. Luego de terminar de procesar esos datos, este registro de estado se pone en ready. para saber el estado del dispositivo se utiliza alguno de los métodos generales de control de E/S:

- Programmed I/O(polling): se utiliza un registro por cada dispositivo y la CPU los monitorea continuamente, cuando detecta un ready, actúa de acuerdo con instrucciones programadas para ese registro en particular.
  - Interrupt-driven I/O(Interrupciones): los dispositivos le dicen a la CPU cuándo tienen datos para enviar. La CPU continúa con otras tareas hasta que un dispositivo solicita una interrupción. Las interrupciones generalmente se habilitan globalmente con un bit en el registro de flags de la CPU llamado interrupt flag (IF) el cual cuando se activa interrumpe el programa que se esté ejecutando, guardando el estado de ese programa y la información variable. Luego, el sistema obtiene el puntero al inicio de la rutina de atención de interrupciones (ISR) correspondiente, y lo carga en el PC. Cuando se termina de ejecutar esta rutina, se reestablece la información que se había guardado para regresar al programa que había sido interrumpido.
  - Direct memory access: es un chip dedicado a hacer polling, aislando la comunicación entre la memoria y los dispositivos de la CPU, permitiendo que la CPU continúe ejecutando otro programa.
-

---

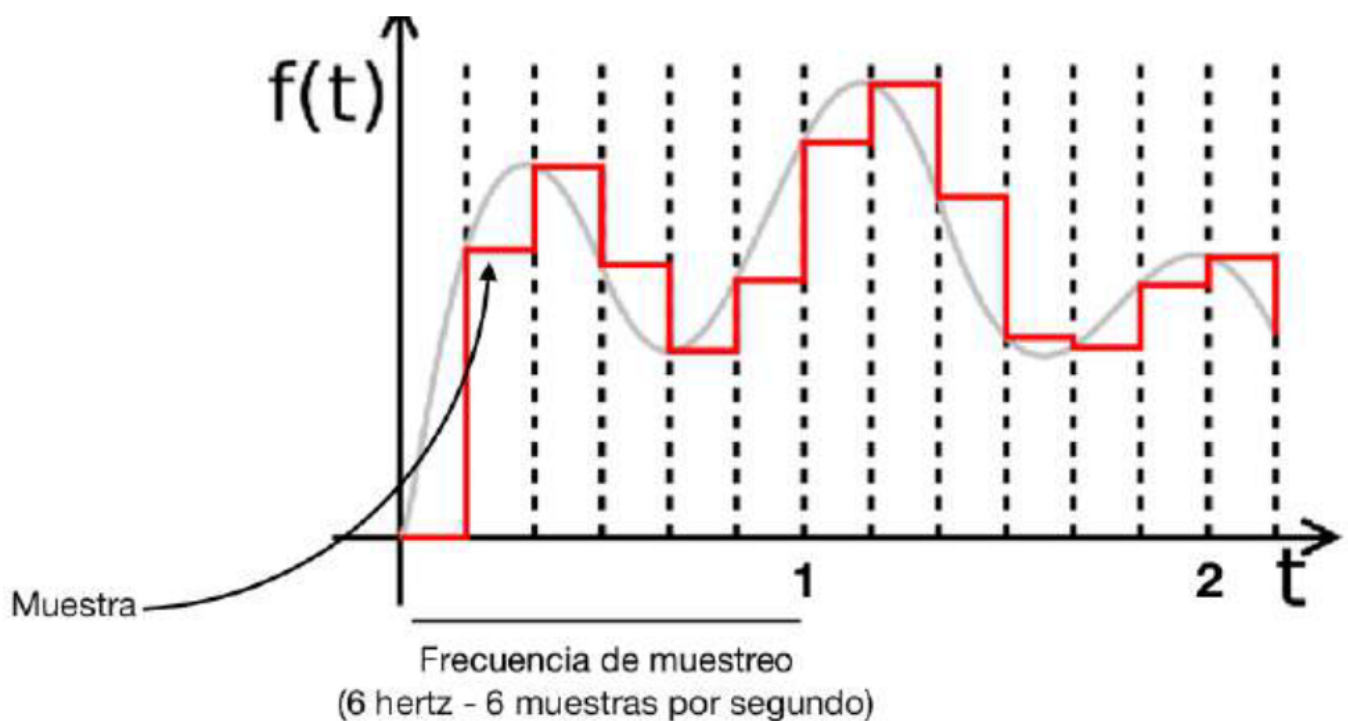
# conversión de señales

Para hacer E/S es necesario a veces hacer una conversión.

## Información analógica

Una señal analógica es un tipo de señal generada por algún fenómeno, con una cierta amplitud y una frecuencia, que es representable por una función continua. Estas señales no pueden ser directamente procesadas por una computadora, sino que es necesario primero modificarlas ya que no todos los valores de estas señales son representables.

Estos problemas se resuelven digitalizando las señales. Esta técnica consiste en limitar la cantidad de posibles valores que pueden tomar las señales, a partir de definir  $n$  umbrales (valor mínimo de una magnitud a partir del cual se produce un efecto determinado), suficientes para capturar la amplitud de las señales, y además fijar una frecuencia de muestreo, que resulte adecuada para no perder valores importantes.



## Información digital

Se toma la información mediante 4 bits (A, B, C, D, de menos significativo a más significativo). La idea es tomar estas 4 señales digitales (0V o 5V) independientes, y asignarles un peso relativo mediante el uso de resistencias (a menor peso, mayor la resistencia), para luego sumarlas, y obtener una señal analógica (en este caso, se obtiene una señal analógica entre 0V y 9.375V).

---